Accelerating lemma learning using joins

LPAR 2008 – Doha, Qatar

Nikolaj Bjørner, Leonardo de Moura
Microsoft Research

Bruno Dutertre
SRI International

Satisfiability Modulo Theories (SMT)

- Arithmetic
- Bit-vectors
- Arrays
- …

Satisfiability Modulo Theories (SMT)

$$\boxed{x+2=y} \Rightarrow f(read(write(a,x,\boxed{3}),\boxed{y-2}) = f(\boxed{y-x+1})$$

Arithmetic

Satisfiability Modulo Theories (SMT)

$$x + 2 = y \Rightarrow f(read(write(a, x, 3), y - 2) = f(y - x + 1)$$

Array Theory

Satisfiability Modulo Theories (SMT)

$$x + 2 = y \Rightarrow \boxed{f}(read(write(a, x, 3), y - 2) = \boxed{f}(y - x + 1)$$

Uninterpreted Functions

SMT: Some Applications @ Microsoft

The Spec# Programming System

HAVOC

ForμLa

Hyper-V

Microsoft Virtualization

Terminator T-2

VCC

SLAM

Yogi

Vigilante

NModel

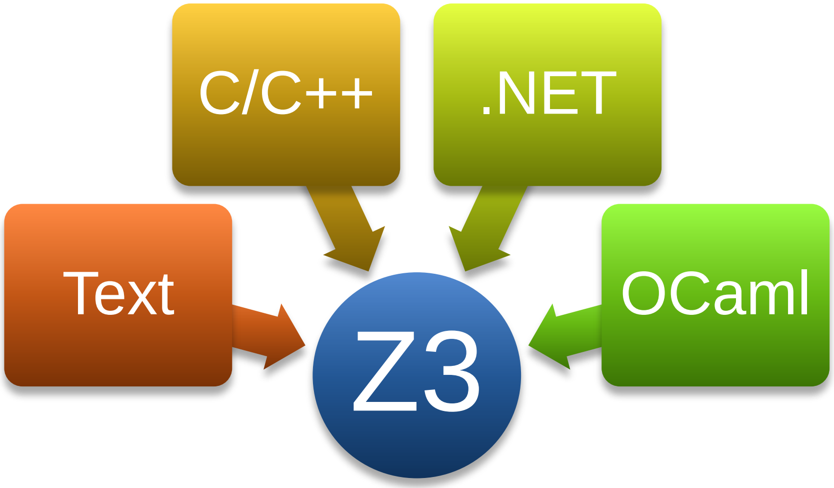
Pex

F7

SAGE

SMT@Microsoft: Solver

- **Z3 is a new solver developed at Microsoft Research.**
- Development/Research driven by internal customers.
- Free for academic research.
- Interfaces:



- http://research.microsoft.com/projects/z3

SMT = DPLL + Theories

$\neg a=b \vee f(a)=f(b), \quad a < 5 \vee a > 10, \quad a > 6 \vee b = 2$

- Guessing (case-splitting)
- Deducing (BCP + Theory propagation)
- Conflict resolution → Backtracking + **Lemma**

Most SMT solvers use only the literals from the given formula!

Is SMT fast???

```
a[0] = 0
if (c1) { a[1] = 0; } else { a[1] = 1; }
...
if (cn) { a[n] = 0; } else { a[n] = 1; }
assert(a[0] == 0);
```

Microsoft Research

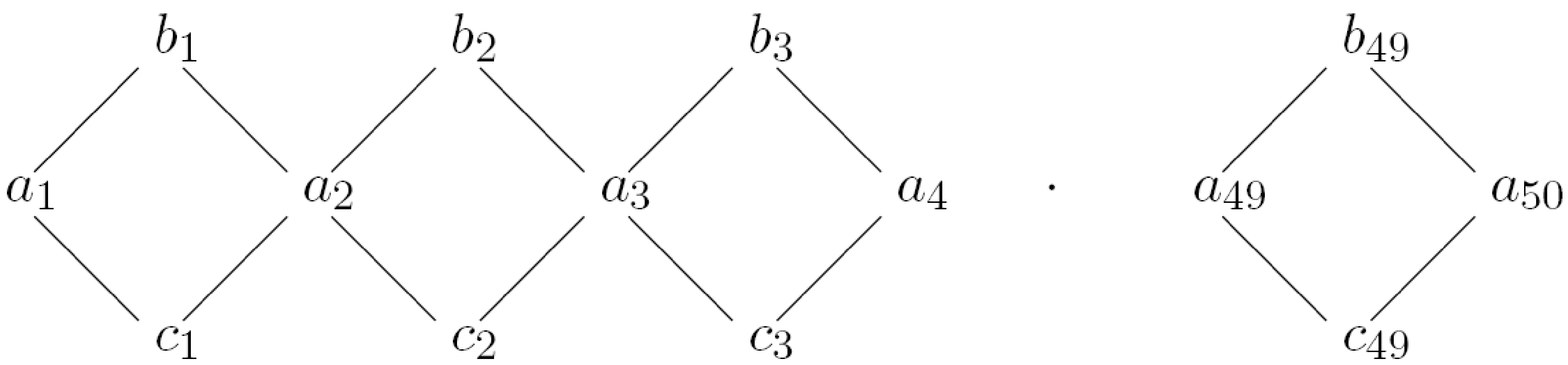
Is SMT fast???

$a_1 = write(a_0, 0, 0)$
$(\neg c_1 \vee a_2 = write(a_1, 1, 0))$
$(c_1 \vee a_2 = write(a_1, 1, 1))$
…
$(\neg c_n \vee a_{n+1} = write(a_n, n, 0))$
$(c_n \vee a_{n+1} = write(a_n, n, 1))$
$read(a_{n+1}, 0) \neq 0$

It takes $O(2^n)$ time if lemmas do not use new literals!

“Diamonds are eternal”

$$a_1 \not\simeq a_{50} \;\wedge\; \bigwedge_{i=1}^{49} [(a_i \simeq b_i \wedge b_i \simeq a_{i+1}) \;\vee\; (a_i \simeq c_i \wedge c_i \simeq a_{i+1})]$$



SP(E) calculus

It can solve “diamonds” in polynomial time.

$$\text{Sup}\ \frac{C \vee a \simeq b \qquad D[a]}{C \vee D[b]} \qquad \text{E-Res}\ \frac{C \vee a \not\simeq a}{C} \qquad \text{E-Fact}\ \frac{C \vee a \simeq b \vee a \simeq c}{C \vee a \simeq b \vee b \not\simeq c}$$

$$\text{Res}\ \frac{C \vee \ell \qquad D \vee \neg\ell}{C \vee D} \qquad \text{Fact}\ \frac{C \vee \ell \vee \ell}{C \vee \ell}$$

The $\mathcal{SP}(E)$ calculus

Very slow in practice!

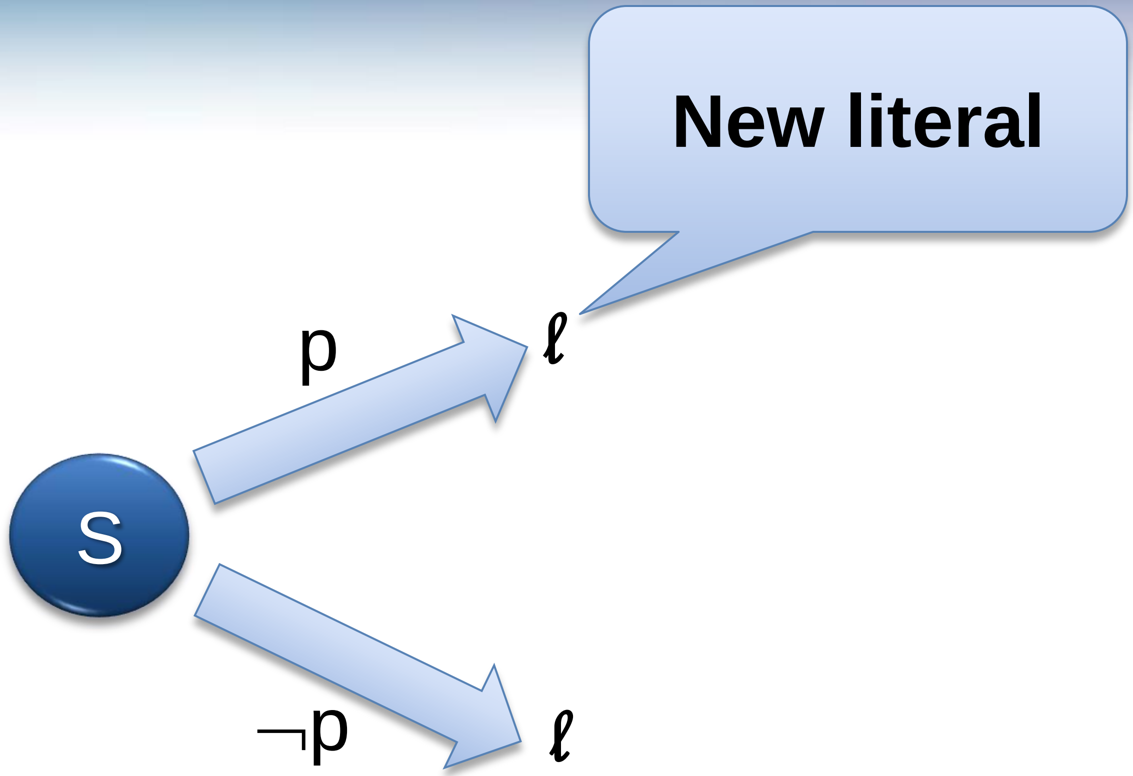
DPLL (E + Δ)

- New literals can be created
  - Case-splitting (guessing)
  - Lemma Learning

Any SP(E) inference can be simulated by DPLL(E+Δ)

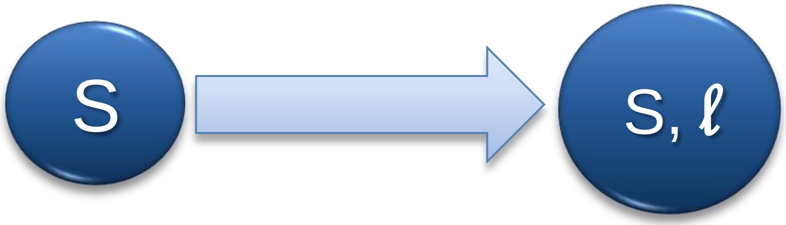
How do we create Δ?

Look ahead



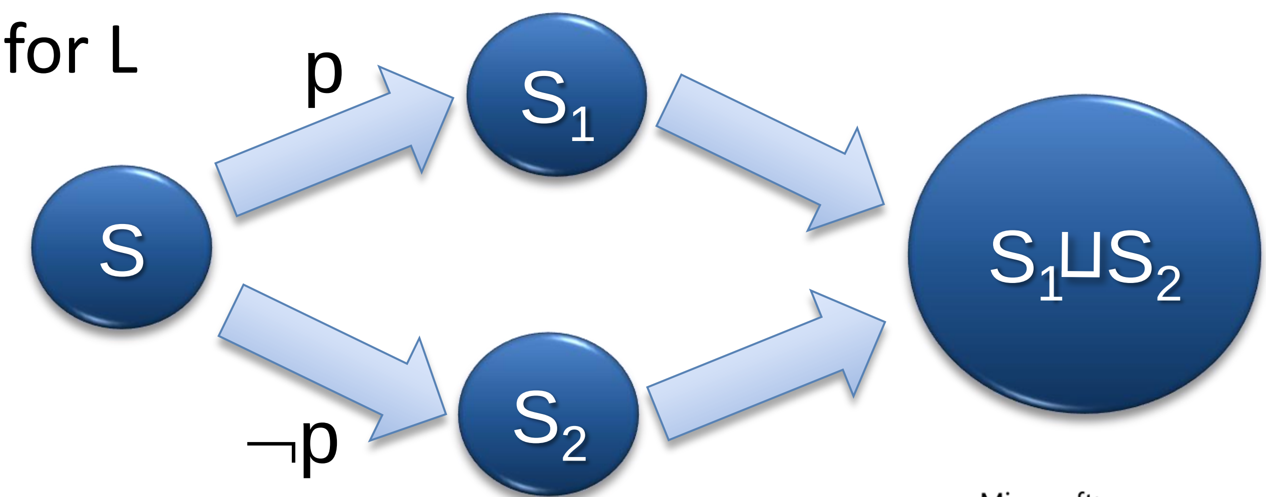
Microsoft Research

Look ahead



"The plan"

- Define language L (of new literals). Examples:
  - (Bounds) $x > 5$
  - (Equality) $x = y$
  - (Difference) $x - y < 3$
- Theory propagation for L
- Join operator for L



Join: Examples (Bounds)

$\neg p \vee q, \quad \neg q \vee x>5, \quad p \vee x>y, \quad y > 4$

Join: Examples (Bounds)

$\neg p \vee q, \quad \neg q \vee x>5, \quad p \vee x>y, \quad y > 4$

p

$\{ x > 5 \}$

Join: Examples (Bounds)

$\neg p \vee q, \quad \neg q \vee x>5, \quad p \vee x>y, \quad y > 4$

p

$\neg p$

$\{ x > 5 \}$

$\{ x > 4 \}$

Join: Examples (Bounds)

$\neg p \vee q, \quad \neg q \vee x>5, \quad p \vee x>y, \quad y > 4$

p

$\neg p$

$\{ x > 5 \}$

$\{ x > 4 \}$

$\{ x > 5 \} \sqcup \{ x > 4 \} = \{x > 4\}$

Join: Examples (Equalities)

$$\{ x = y, y = z, x = z \} \sqcup \{ x = z, z = w, x = w \} = \{x = z\}$$

Join: Examples (Difference constraints)

$$\{ x - y < 3 \} \sqcup \{ x - y < 2, y - z < 1, x - z < 3\} = \{x - y < 3\}$$

Join

- Other examples:
  - Linear arithmetic: polyhedral.
  - Array partial equalities:

    $a =_i b \quad (\text{forall } x: x = i \vee a[x] = b[x])$
- k-look ahead.

Conclusion

- SMT solvers are fast, but they may choke in simple formulas.
- DPLL(join) = SMT + "Abstract Interpretation".
- Future work: new literals during conflict resolution.
- http://research.microsoft.com/projects/z3

Thank You!